2012-08-02

Hierarchical Task Networks in COMBATXXI, Bringing AI Techniques Into Closed Form Combat Simulations

Reeves, David

Monterey, California: Naval Postgraduate School.

Hierarchical Task Networks in COMBATXXI

Bringing AI Techniques Into Closed Form Combat Simulations

David Reeves & Imre Balogh

Agenda

- Introduction & Motivation
- Hierarchical Task Networks (HTN)
- HTNs in Dynamic Environments
- HTNs in COMBATXXI
- Future Work

Introduction

- Goal of Research:
  - Develop better way to build complex behaviors in combat models
  - Target simulations are analytical combat simulations
    - Techniques apply to any simulation where autonomous agents are used
- Background:
  - Analytical Simulations
    - System/TTP evaluation
    - Tend to be closed form – no user interaction during runs
      - → Highly Scripted
      - → or Autonomous Agents
  - COMBATXXI
    - Analytical Sim used by the Army and Marines
    - Stochastic, entity level, event stepped, closed form model
  - Sponsored by Marine Corps Combat Development Command

Hierarchical Task Networks

- HTNs are a type of automated planning system in which dependency between actions/tasks is expressed in the form of networks or trees
- The planning process starts with a high-level description of the problem as a set of goal tasks
- To accomplish the goal tasks, a plan is defined through action decomposition
- A set of sub-tasks is produced that must be sequenced appropriately to complete the higher-level task while accounting for constraints
- Sub-tasks may need further decomposition
- This process is repeated recursively until the sub-tasks are no longer decomposable.

HTN Tasks

- Primitive Tasks – Tasks that require no further decomposition to be performed
- Compound Tasks – Tasks that need to be decomposed into a simpler set of tasks
- Goal Tasks – Tasks that, when completed, corresponds to the goal of the network or tree

A goal task corresponds to the achievement of the desired world state

- Constraint Node – condition to control network branch to be executed

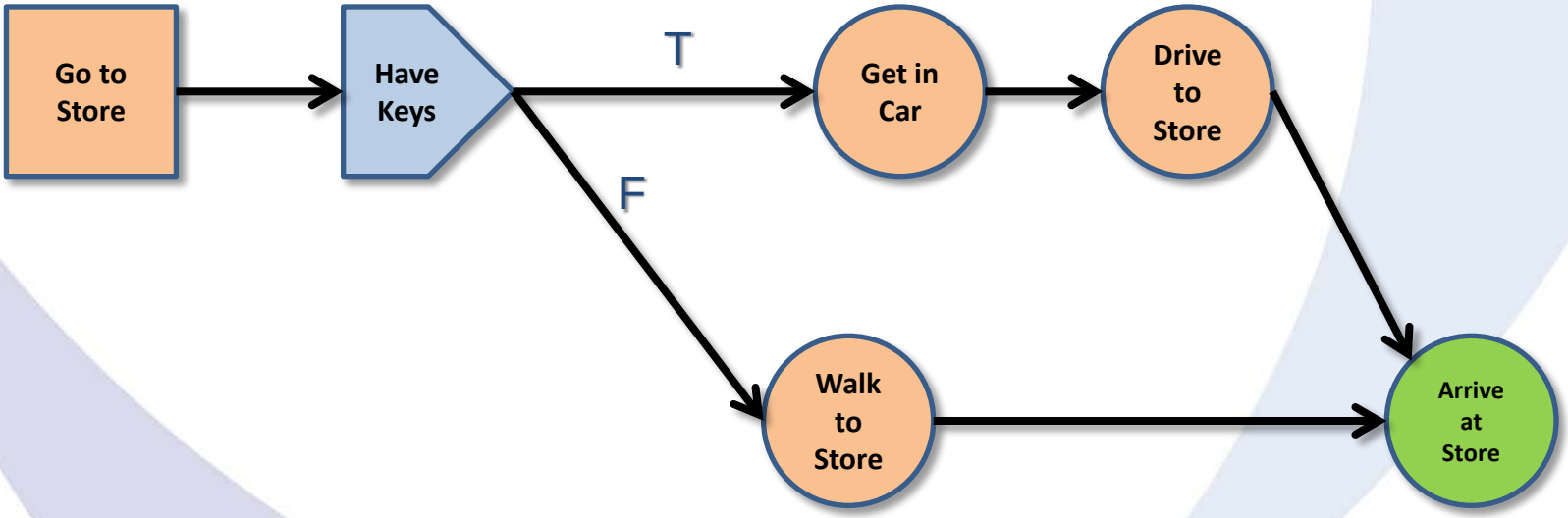
HTN Example: Go to Store

START TREE (L to R)

- 'Go To Store' task starts plan generation
- 'Go To Store' (compound task) passes to the first node- 'have Keys'

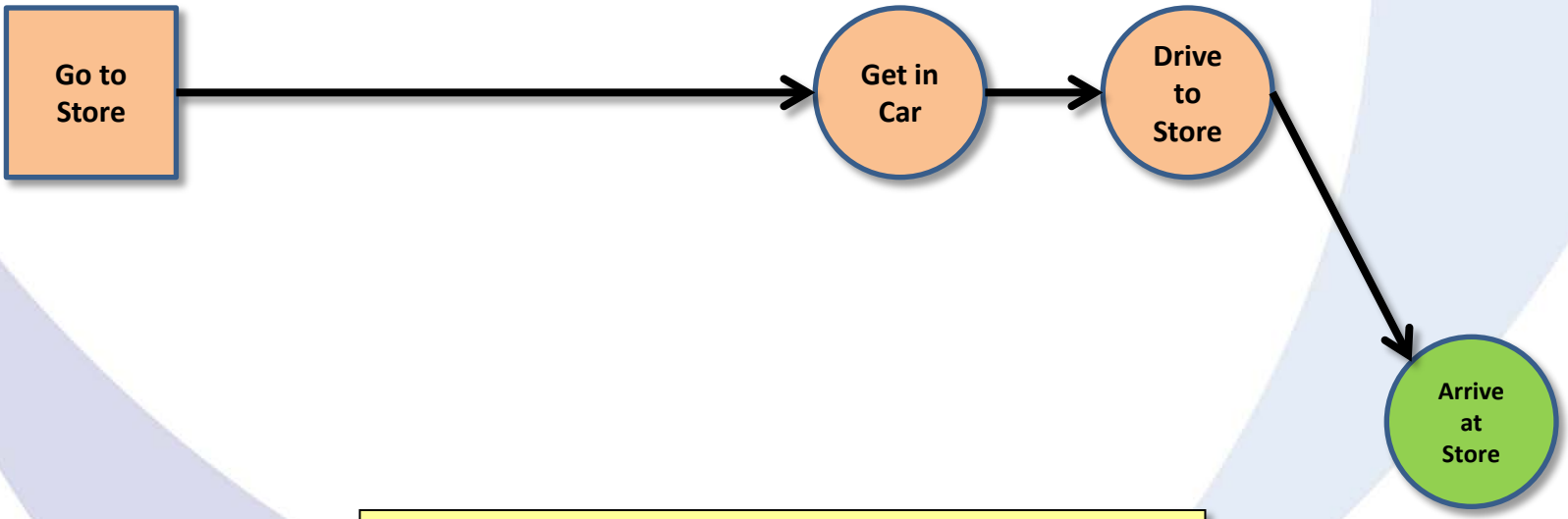
PASSES CONSTRAINT

- 'Get In Car' (primitive task) executes and passes to 'Drive To Store'.
- 'Drive To Store' (primitive task) executes and passes to 'Arrive At Store'.
- 'Arrive At Store' (goal task) completes the plan



HTN Example: Go to Store

Resulting Plan

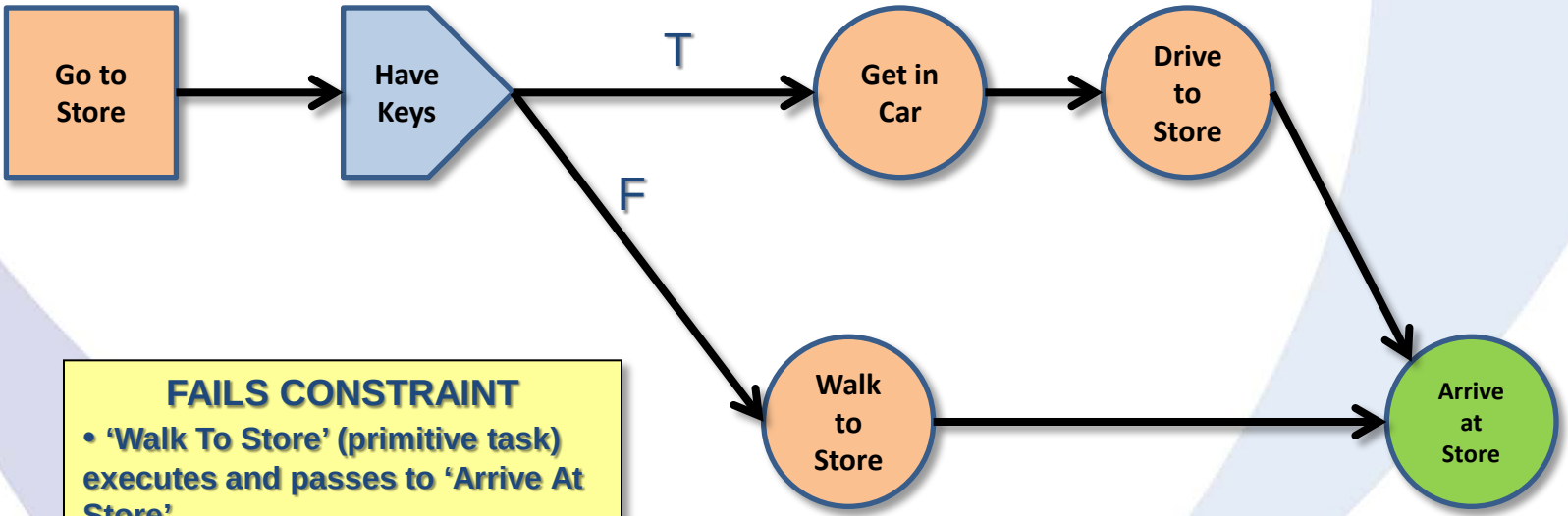


Plan execution system executes the plan

HTN Example: Go to Store

START TREE (L to R)

- 'Go To Store' task starts plan generation
- 'Go To Store' (compound task) passes to the first node- 'have Keys'

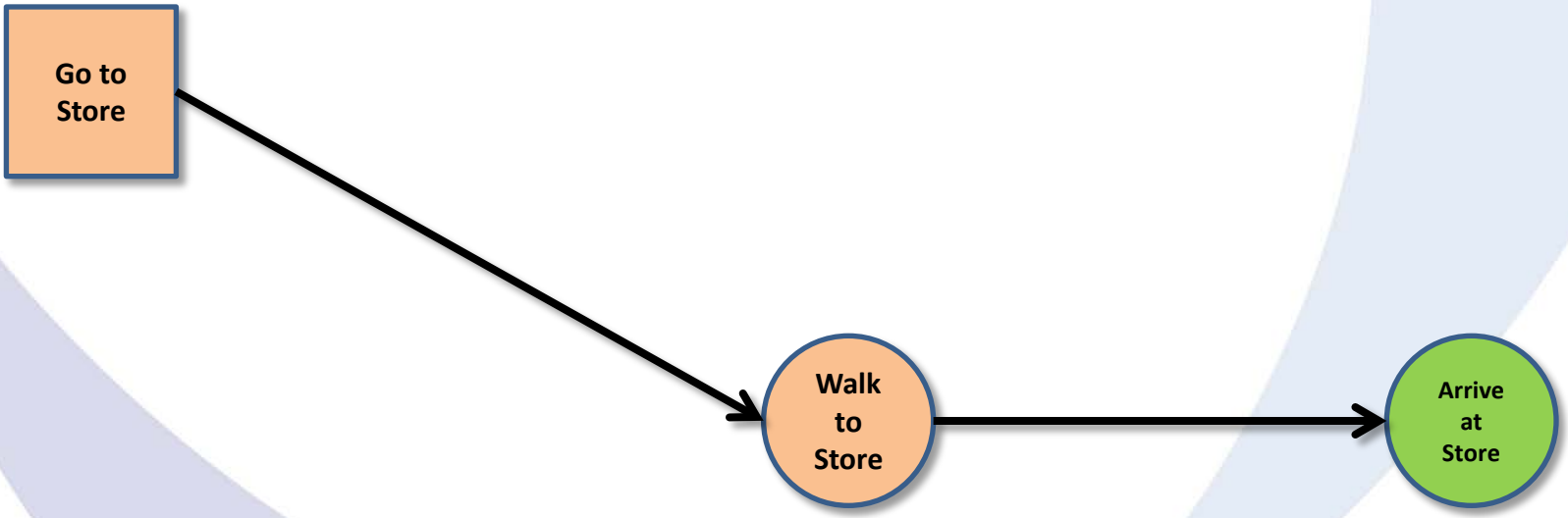


FAILS CONSTRAINT

- 'Walk To Store' (primitive task) executes and passes to 'Arrive At Store'
- 'Arrive At Store' (goal task) completes the plan

HTN Example: Go to Store

Resulting Plan



Plan execution system executes the plan

HTNs in Dynamic Environments

- An HTN driven plan is based on values of the constraints at the time of plan formulation
  - Not a problem in static environments
- In a dynamic environment, there is no guarantee that the plan can be completed just because it was possible to generate it
- To account for changing environments, a method is needed to review the validity of the plan as it is being executed
- Generating plans can be expensive, so regeneration needs to be limited if possible

“Replan” Events

- In event-stepped simulations, entities can be aware of the events that can affect their state
- By selectively listening to events that could affect their plans, entities can control how often they must regenerate their plans
- To take advantage of this we introduce a new type of event, the “replan event”, which will cause the entity to suspend the execution of the current plan and reevaluate the HTN being used
- In the process of defining an HTN, part of the specification is the set of events that can generate a replan event

HTNs in COMBATXXI

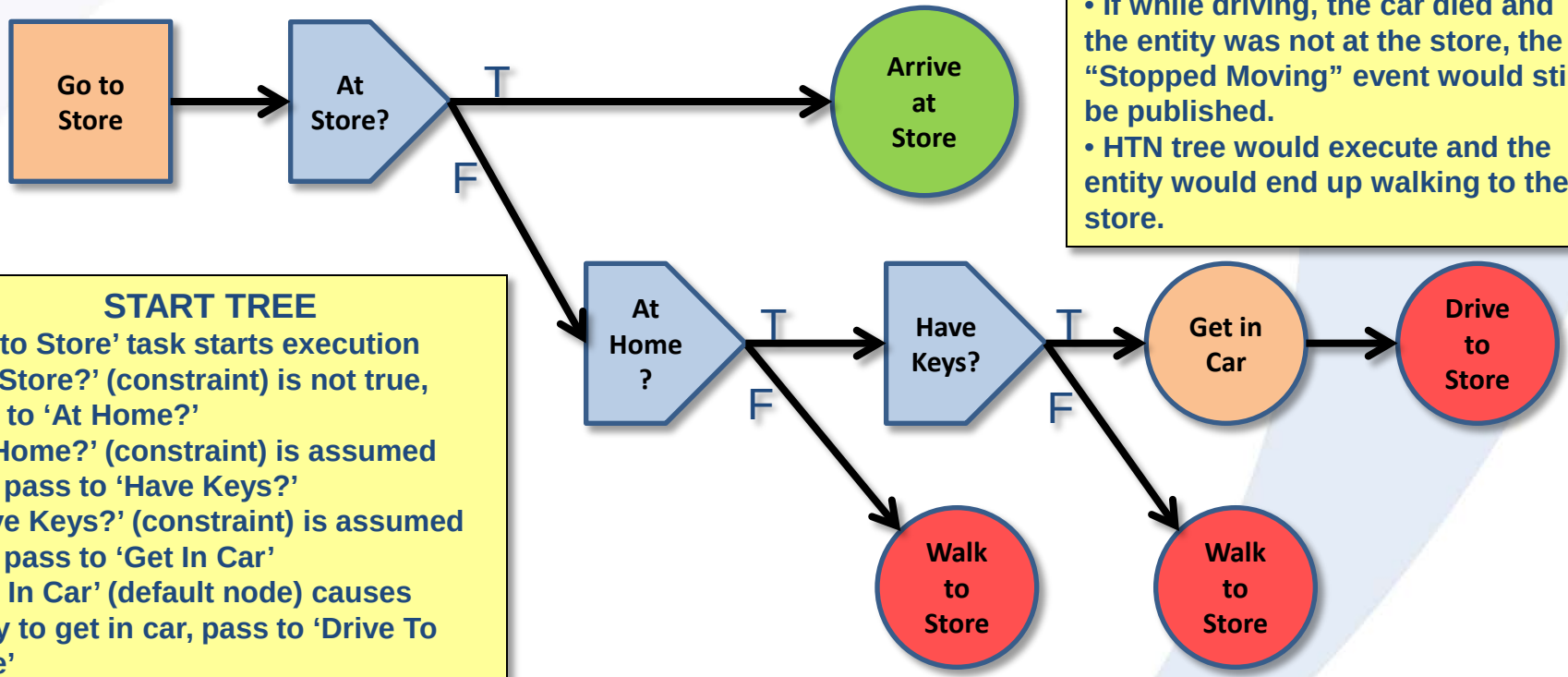
- A COMBATXXI HTN tree is made up of nodes arranged in a network with constraints that determine which branches of the tree are processed as the plan is run
- There are three types of nodes in a COMBATXXI HTNs
  - Default Node: This is a basic node. No special actions are taken when the planner encounters a default node
  - Interrupt Node: An interrupt node tells the planner that it has reached a node for which it should no longer continue processing nodes. The plan is "interrupted" and stops, but is not finished
  - Goal Node: A goal node is a node that tells the planner that it's reached a spot for which the world state is now in the desired state. The plan is finished and will be removed

Interrupt Goal Task

- An interrupt goal represents a task that, when completed, will have the world state be closer to our final goal state, but is not the final goal state of the entire HTN
- Interrupt goal tasks serve two purposes:
  - Represent tasks that takes some time to complete and are likely places where the plan may be interrupted
  - Allow for the "lazy generation" of the plan – only part of plan needs to be generated
- Regeneration of plans are tied to specific events
- Interrupt goal are represented by red circles

‘Go to Store’ in COMBATXXI

- **Replan event is “Stopped Moving”**



UNEXPECTED EVENT TRIGGERED

- If while driving, the car died and the entity was not at the store, the “Stopped Moving” event would still be published.
- HTN tree would execute and the entity would end up walking to the store.

START TREE

- ‘Go to Store’ task starts execution
- ‘At Store?’ (constraint) is not true, pass to ‘At Home?’
- ‘At Home?’ (constraint) is assumed true, pass to ‘Have Keys?’
- ‘Have Keys?’ (constraint) is assumed true, pass to ‘Get In Car’
- ‘Get In Car’ (default node) causes entity to get in car, pass to ‘Drive To Store’
- ‘Drive To Store’ (interrupt node) executes and the planner will stop but the tree is not yet finished

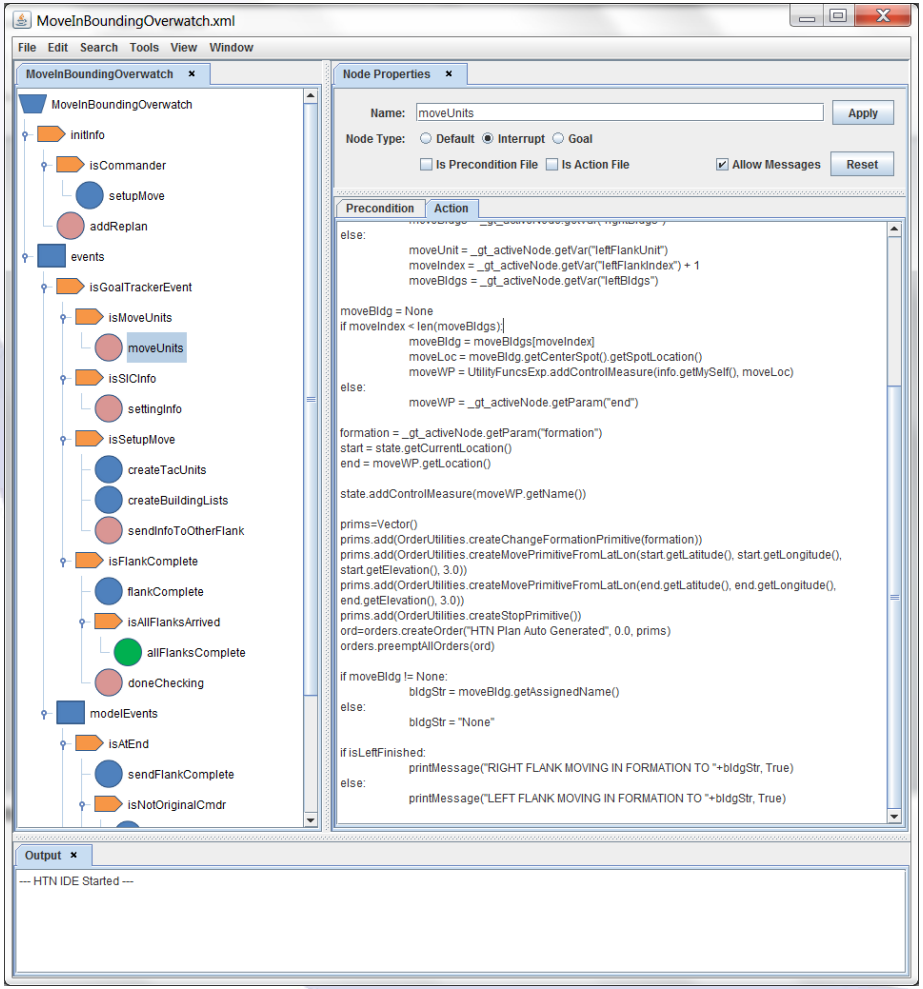
“STOP MOVING” EVENT TRIGGERED

- ‘Go to Store’ task starts execution ‘Go to Store’ task starts execution
- ‘At Store?’ is true, pass to ‘Arrive At Store’
- ‘Arrive At Store’ (goal node) indicates the world is at the desired state
- HTN is finished and the plan is removed.

HTN Portability & Re-use

- To support building modular and reusable HTNs, parameters were added to the HTNs in COMBATXXI
  - Allows for creation of HTNs that can be used in multiple contexts
- With the introduction of interrupt nodes and HTN controlled replans, an HTN needs to be able to set and query status information accumulated as it executes
  - Status information can then be used in preconditions or tasks to affect the flow of the plan and execution of its tasks
  - To support this, all HTNs have their own “variable space” attached
  - Tasks are allowed to set and query variables in this variable space

Graphical User Interface



- Full function Graphical HTN developed
  - Tailored for COMBATXXI
- Extensive helper functionality to help user build actions
- Supports saving parts of HTN to support reuse
- Based in interface metaphors used in common programming IDEs such as NetBeans and Eclipse

Use in Recent Analysis

- This HTN methodology was recently rigorously tested in a complex scenario co-developed at NPS and the Marine Corps Combat Development Command
- This scenario was used to support analysis of the Amphibious Combat Vehicle (ACV)
- The scenario was composed of over 4000 entities executing complex maneuvers to include an amphibious landing, movement to contact, a bridge crossing, and a final attack supported by vehicles in a support by fire position
- HTNs were used to model each of these complex maneuvers and numerous others
- This approach allowed us to raise dynamic behaviors in COMBATXXI to a new level of sophistication

Future Work

- Continue to refine this methodology based on experiences with using it to support the ACV AoA
- Develop ways that support building “reusable” HTNs that will help define libraries of HTN based behaviors
- Investigate extending HTNs to support a notion of “inheritance” that will allow for the definition of abstract HTNs
- Investigating machine learning (e.g. reinforcement learning) can be used with HTNs